

Online Learning with Regularized Kernel for One-Class Classification

Chandan Gautam*, Aruna Tiwari, Sundaram Suresh, *Senior Member, IEEE*, and Kapil Ahuja

Abstract—This paper presents an online learning with regularized kernel based one-class extreme learning machine (ELM) classifier and is referred as “online RK-OC-ELM”. The baseline kernel hyperplane model considers whole data in a single chunk with regularized ELM approach for offline learning in case of one-class classification (OCC). Further, the basic hyper plane model is adapted in an online fashion from stream of training samples in this paper. Two frameworks viz., boundary and reconstruction are presented to detect the target class in online RK-OC-ELM. Boundary framework based one-class classifier consists of single node output architecture and classifier endeavors to approximate all data to any real number. However, one-class classifier based on reconstruction framework is an autoencoder architecture, where output nodes are identical to input nodes and classifier endeavor to reconstruct input layer at the output layer. Both these frameworks employ regularized kernel ELM based online learning and consistency based model selection has been employed to select learning algorithm parameters. The performance of online RK-OC-ELM has been evaluated on standard benchmark datasets as well as on artificial datasets and the results are compared with existing state-of-the-art one-class classifiers. The results indicate that the online learning one-class classifier is slightly better or same as batch learning based approaches. As, base classifier used for the proposed classifiers are based on the ELM, hence, proposed classifiers would also inherit the benefit of the base classifier i.e. it will perform faster computation compared to traditional autoencoder based one-class classifier.

Index Terms—One-Class Classification, Kernel, Regularization, Online Sequential, Extreme Learning Machine (ELM), Outliers Detection.

I. INTRODUCTION

THE term ‘One-Class Classification’ (OCC) was coined by Moya et al. [1]. The OCC has been generally employed to solve the problem of novelty, outlier, intrusion or fault detection [2]. These problems are also solved by multi-class classification when samples of both classes, normal and outlier class, are available [3] [4] [5] [6]. However, the OCC has been applied when either data of only one class is available or the data belonging to other classes is very rare. One-class classifier simply describes the data, therefore it is also called as a data descriptor. Most of the research in OCC has been based upon Support Vector Machine (SVM) and

neural network. Two types of SVM based one-class classifier have been proposed, namely, One-Class SVM (OCSVM) [7] and Support Vector Domain Description (SVDD) [8]. After surveying various papers, it has been observed that SVM based one-class classifier has been explored more compared to other one class methods [2]. SVM based one-class classifier has also been developed for incremental and online learning [9]. For various one-class classifiers like SVDD, autoencoder etc., a toolbox is available in [10]. Based on different nature of data used during learning hyperplane of OCC, it can be classified as [11] (i) with positive examples only, (ii) with the positive and small amount of negative examples only, and (iii) with positive and unlabeled data. Online learning has attracted researchers in recent years due to its capability to handle high volume of streaming data. This is because it is computationally expensive to handle as well as costly to store large volumes of data.

Neural network with backpropagation has been explored in past for OCC task [12] [13]. The learning algorithm in these one-class classifiers uses iterative and computationally intensive gradient descent based approach. This issue has been addressed by single layer feed-forward network in past decades. Single layer feed-forward network got quite attention by researchers for multi-class classification and regression due to its fast training capability as it does not adjust the weights by back propagation. Huang et al. [14] [15] also developed a single layer feed-forward network, called as Extreme Learning Machine (ELM). Online version of ELM has also been developed for multi-class classification and regression tasks with random [16] and kernel feature mapping [17] [18] [19]. In past decades, ELM has been well expanded in both dimensions: theory [20] [21] [22] [23] and application [24] [18] [25] [26]. A detailed survey on ELM and its application can be found in [27]. Recently, Leng et al. [28] developed ELM for OCC due to its fast learning speed, which supports only offline learning. Leng et al. [28] tested their model with only one type of threshold deciding criteria, i.e. rejection of few percentages of most deviant training samples after completion of training. Later, Iosifidis [29] improvised ELM based one-class classifier based on geometric information of class for offline learning. Most recently, Gautam et al. [30]¹ [31] explored further ELM based one-class classifier for mainly offline learning and tested it with three different threshold criteria. Gautam et al. [30] have shown particularly two aspects: (i) random feature mapping based offline methods have been outperformed by kernel feature mapping based offline methods (ii) the possibility of

*Corresponding Author

Chandan Gautam is with the Indian Institute of Technology Indore, (e-mail: chandangautam31@gmail.com).

Aruna Tiwari is with the Indian Institute of Technology Indore, (e-mail: artiwari@iiti.ac.in).

Sundaram Suresh is with the Nanyang Technological University, 639798 Singapore, (e-mail: ssundaram@ntu.edu.sg).

Kapil Ahuja is with the Indian Institute of Technology Indore, (e-mail: kahuja@iiti.ac.in).

¹you can download the submitted version of accepted paper form the link: <http://goo.gl/XhkVpE>. We are providing this url because paper is accepted but not published yet.

development of online learning with ELM based one-class classifier. However, developed online one-class classifiers produce very inferior performance compared to offline one-class classifiers. Hence, there is a need to develop an online one-class classifier with fast learning approach, which is required to provide performance similar to batch learning algorithm and handle stream of data. Therefore, regularized kernel based online-sequential one-class classifiers have been proposed in this paper, which can perform similar to offline one-class classifiers. Proposed one-class classifiers are developed for two types of framework: (i) boundary and (ii) reconstruction. Boundary framework based proposed method is developed as a fast online single output node architecture with kernel feature mapping and reconstruction based approach is developed as a fast online autoencoder with kernel feature mapping. Boundary framework based proposed method is explored with one type of threshold and reconstruction framework based proposed method is explored with two type of threshold criteria. Out of two threshold criteria, one rejects few samples from target data to decide threshold point, however, another threshold criteria doesn't reject any sample from target data to decide threshold point. Data description capability of the proposed methods is tested on synthetic or artificial datasets and compared the outcomes with random feature mapping based online one-class classifier. Results on six benchmark data exhibit that performance of the proposed online one-class classifiers is either similar or better compared to traditional offline autoencoder and incremental SVDD. Among two types of framework, reconstruction framework based proposed methods perform better than boundary framework based proposed methods.

The rest of the paper is organized as follows: Section II discussed about the proposed work. Performance evaluation has been discussed in Section III. Section IV contains the conclusion of the work.

II. ONLINE LEARNING WITH REGULARIZED KERNEL FOR OCC

In this section, ELM based online sequential one-class classifier is modeled by taking both factors viz., regularization and kernelization, into account. This is called as online learning with regularized kernel for one-class ELM (**Online RK-OC-ELM**) in this paper. Regularization factor helps classifier to achieve a better generalization capability for noisy data. Tikhonov regularization [32] has been employed due to its capability to handle ill-posed and singular problem, which generally emerges in solving the inverse problem. Following subsections discusses for two types of framework viz., boundary and reconstruction:

A. Online RK-OC-ELM: Boundary Framework Based Approach

In Boundary framework based **Online RK-OC-ELM**, model is trained by only target data X and endeavored to approximate all data to any real number. Fig. 1 shows online OCC with single output node architecture. In Fig. 1, given a stream of training data X , $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \dots, (\mathbf{x}_t, c_t), \dots\}$, where $\mathbf{x}_t = [x_t^1, x_t^2, \dots, x_t^n] \in \mathbb{R}^n$ is n -dimensional input of

the t^{th} sample and c_t is the class label of the target class, which is same for all the training data. Input layer takes data for t^{th} input sample is coded as (\mathbf{x}_t, R_t) because model has to approximate all data to any real number R . Target output vector \mathbf{R} is represented as $[R_1, R_2, \dots, R_t, \dots]$, however, value of R_t will be same for all samples. Here, value of R_t is considered as 1 for all the experiments. Further, kernel feature mapping has been employed between input and hidden layer ϕ . Different symbol has been used for kernel matrix to avoid confusion between kernel ϕ and random feature mapping \mathbf{H} . During training, hidden layer output or kernel matrix ϕ will be a square symmetric matrix of size $[t \times t]$. Output weight β for any t samples in sequence of data is represented as $[\beta_{11}, \beta_{21}, \dots, \beta_{(t-1)1}, \beta_{tt}]$. $\hat{\mathbf{R}} = [\hat{R}_1, \hat{R}_2, \dots, \hat{R}_t, \dots]$ is the predicted output vector and \hat{R}_t is the predicted output for t^{th} sample. $\hat{\mathbf{c}} = [\hat{c}_1, \hat{c}_2, \dots, \hat{c}_t, \dots]$ is the predicted class vector, where, \hat{c}_t is the predicted class for t^{th} sample. Online learning algorithm of **Online RK-OC-ELM** for boundary framework based approach is as discussed below. Result of breast cancer and ecoli dataset is provided jointly in the table as both have well separated classes. Similarly, result of diabetes and liver datasets have provided jointly because both have overlapped classes, and, sonar and spectf datasets are also provided jointly as they are high dimensional datasets.

For initial chunk of N_0 dataset $\{\mathbf{X}_0, \mathbf{R}_0\}$, objective is to minimize the output weight β_0 as well as error E_0 between expected (\mathbf{R}_0) and predicted value ($\phi(\mathbf{X}_0)\beta_0$). For regularization, λ is used as a regularization parameter with minimization problem. Hence, minimization problem can be written as follows:

$$\begin{aligned} \text{Minimize : } L &= \frac{1}{2} \|\beta_0\|^2 + \lambda \frac{1}{2} \|\mathbf{E}_0\|^2 \\ \text{Subject to : } \phi(\mathbf{X}_0)\beta_0 &= \mathbf{R}_0 - \mathbf{E}_0 \end{aligned} \quad (1)$$

After solving above minimization problem using Karush-Kuhn-Tucker (KKT) theorem [33], it yields the following solution:

$$\begin{aligned} \beta_0 &= \mathbf{P}_0 \mathbf{R}_0 \\ \mathbf{P}_0 &= \phi_0^{-1} \\ \phi_0 &= \phi(\mathbf{X}_0) \end{aligned} \quad (2)$$

Here, ϕ_0 is a kernel matrix for initial N_0 of size $N_0 \times N_0$. Kernel matrix ϕ_0 is defined based on Mercer's condition. Hence, any kernel method which satisfies Mercer's condition can be adopted as the kernel for the proposed classifier. For initial N_0 sample, kernel matrix will be defined as:

$$\phi = \phi_0 = \left(\begin{bmatrix} \Omega_{11} & \Omega_{12} & \dots & \Omega_{1N_0} \\ \Omega_{21} & \Omega_{22} & \dots & \Omega_{2N_0} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{N_01} & \Omega_{N_02} & \dots & \Omega_{N_0N_0} \end{bmatrix} + \frac{I}{\lambda} \right) \quad (3)$$

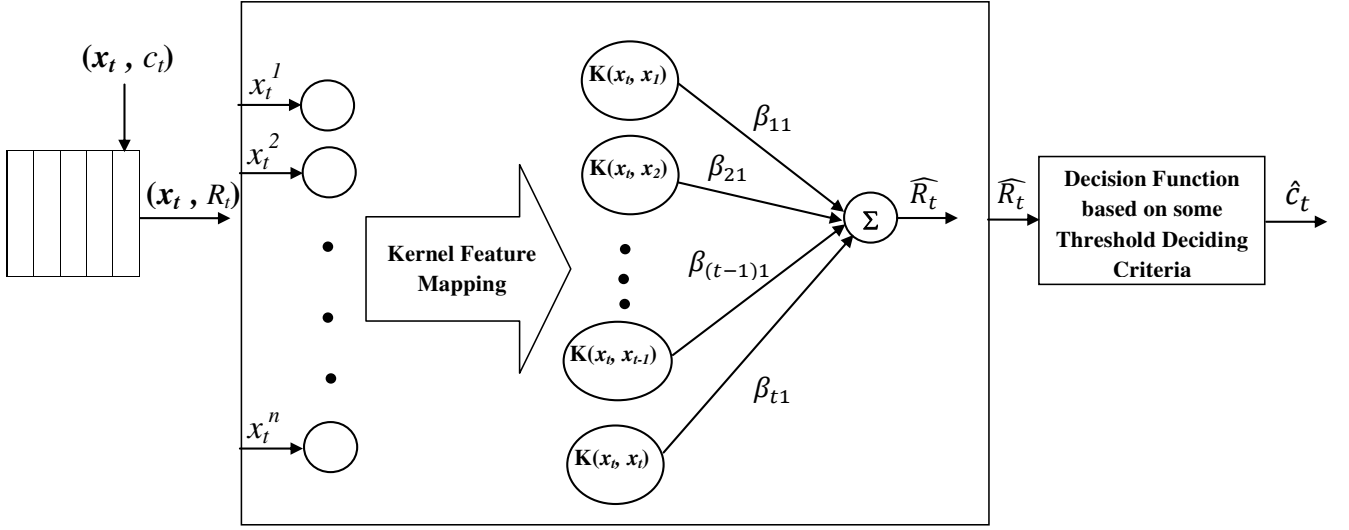


Fig. 1: Schematic Diagram of Online Sequential One Class Extreme Learning Machine: Boundary Based

$$P = P_0 = \phi_0^{-1} = \left(\begin{bmatrix} \Omega_{11} & \Omega_{12} & \dots & \Omega_{1N_0} \\ \Omega_{21} & \Omega_{22} & \dots & \Omega_{2N_0} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{N_01} & \Omega_{N_02} & \dots & \Omega_{N_0N_0} \end{bmatrix} + \frac{I}{\lambda} \right) \quad (4)$$

Initially, ϕ and P will be equal to ϕ_0 and P_0 respectively. Here, ϕ represents kernel matrix and P represents inverse of this kernel matrix for all the arrived samples till now for training. Here, ϕ and P will be updated continuously for any upcoming new samples X^v as per Equation (5), where, $X^v = \{(x_1^v, c_1), (x_2^v, c_2), \dots, (x_s^v, c_s)\}$ and $X^v \subset X$. X^v is just next chunk of the data. Current value of ϕ is represented as ϕ_u . Here, u and v simply denote old and new values respectively. Now, calculate ϕ after arrival of new sample as follows:

$$\phi = \begin{bmatrix} \phi_u & \phi_{u,v} \\ (\phi_{u,v})^T & \phi_v \end{bmatrix} \quad (5)$$

Here, ϕ is combination of four block matrices. ϕ_u is the old value of ϕ . Block matrix $\phi_{u,v}$ and ϕ_v in Equation (5) are calculated as per Equation (6), which is discussed below.

Let the number of samples processed till now be b and number of samples in the current chunk be s . b is initially equal to N_0 . Update b and s each time when calculation starts for new samples. The block matrices ϕ_v and $\phi_{u,v}$ can be defined as follows:

$$\phi_v = \left(\begin{bmatrix} K(x_1^v, x_1^v) & \dots & K(x_1^v, x_s^v) \\ \vdots & \ddots & \vdots \\ K(x_s^v, x_1^v) & \dots & K(x_s^v, x_s^v) \end{bmatrix} + \frac{I}{\lambda} \right) \quad (6)$$

$$\phi_{u,v} = \begin{bmatrix} K(x_1^u, x_1^v) & \dots & K(x_1^u, x_s^v) \\ \vdots & \ddots & \vdots \\ K(x_b^u, x_1^v) & \dots & K(x_b^u, x_s^v) \end{bmatrix} \quad (7)$$

Now, value of P will be inverse of ϕ in Equation (5) as follows:

$$P = \phi^{-1} = \begin{bmatrix} \phi_u & \phi_{u,v} \\ (\phi_{u,v})^T & \phi_v \end{bmatrix}^{-1} \quad (8)$$

Further, compute the inverse in Equation (8) using block matrix inverse formula [34].

$$S = D^{-1} = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}^{-1} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \quad (9)$$

where,

$$\begin{aligned} S_{11} &= (D_{11} - D_{12}D_{22}^{-1}D_{21})^{-1} \\ S_{12} &= -D_{11}^{-1}D_{12}(D_{22} - D_{21}D_{11}^{-1}D_{12})^{-1} \\ S_{21} &= -D_{22}^{-1}D_{21}(D_{11} - D_{12}D_{22}^{-1}D_{21})^{-1} \\ S_{22} &= (D_{22} - D_{21}D_{11}^{-1}D_{12})^{-1} \end{aligned}$$

Hence, Equation (8) will be rewritten as follows:

$$P = \phi^{-1} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \quad (10)$$

P_{11} , P_{12} , P_{21} and P_{22} in Equation (10) can be written as follows:

$$\begin{aligned} P_{11} &= (\phi_u - \phi_{u,v}\phi_v^{-1}\phi_{u,v}^T)^{-1} \\ P_{12} &= -\phi_u^{-1}\phi_{u,v}P_{22} \\ P_{21} &= -\phi_v^{-1}\phi_{u,v}^TP_{11} \\ P_{22} &= (\phi_v - \phi_{u,v}^T\phi_u^{-1}\phi_{u,v})^{-1} \end{aligned} \quad (11)$$

P_{11} can be expanded by employing Woodbury formula [34] as:

$$\begin{aligned} P_{11} &= (\phi_u - \phi_{u,v}\phi_v^{-1}\phi_{u,v}^T)^{-1} \\ &= \phi_u^{-1} - \phi_u^{-1}\phi_{u,v}(\phi_{u,v}^T\phi_u^{-1}\phi_{u,v} \\ &\quad + \phi_v^{-1})^{-1}\phi_{u,v}^T\phi_u^{-1} \end{aligned} \quad (12)$$

In a similar fashion, Equation (11) can be explored for P_{12} , P_{21} and P_{22} . After processing the training dataset X , output function can be written for any set of k samples $X_k = \{x_1, x_2, \dots, x_k\}$ as follows:

$$f(X_k) = \begin{bmatrix} K(X, x_1) \\ \vdots \\ K(X, x_k) \end{bmatrix}^T PR \quad (13)$$

where, $K(X, x_i)$ denotes kernel vector for i^{th} sample x_i . Further, perform the following steps to decide whether any sample is outlier or not:

- (i) Calculate distance (d) between the predicted value of the t^{th} training sample and R as follows:

$$d(x_t) = |f(x_t) - R_t| = |\hat{R}_t - R_t| \quad (14)$$

- (ii) After calculating distances d as per above Equation (14), sort the differences in decreasing order. Further, reject few percent of training samples based on the deviation. More deviated sample will be rejected first because they are most probably far from distribution of the target data. The threshold will be decided based on those deviations as follows:

$$\theta_1 = d(\lfloor \eta * N \rfloor) \quad (15)$$

Where $0 < \eta \leq 1$, N is the number of training samples and η is the fraction of rejection of training samples for deciding threshold value. We have considered 10% of rejection, i.e. $\eta = 0.1$. Now, generate a decision function to decide whether any new sample z belongs to the target or outlier, where $z = [z^1, z^2, \dots, z^n]$ as per following equation:

$$Sign(\theta_1 - d(z)) = \begin{cases} 1, & z \text{ is classified as target} \\ -1, & z \text{ is classified as outlier} \end{cases} \quad (16)$$

The above proposed approach is summarized as pseudo code in **Algorithm 1**.

B. Online RK-OC-ELM: Reconstruction Based Approach

In reconstruction framework based **Online RK-OC-ELM**, model is trained by only target data X and endeavored to approximate all data to itself. Fig. 2 shows the architecture for reconstruction framework based **Online RK-OC-ELM**. In Fig. 2, given a stream of training data X is defined same as discussed above in the boundary framework based one-class classifier. Input layer takes data for t^{th} input sample as (x_t, x_t) because the target is identical to the input layer. Further, kernel feature mapping is employed between input and hidden layer ϕ . During training, hidden layer output or kernel matrix ϕ will be a square symmetric matrix of size $[t \times t]$. Output weight β till t^{th} samples is represented as $[\beta_1, \beta_2, \dots, \beta_{(t-1)}, \beta_t]$. $\hat{X} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_t, \dots)$ is the predicted output vectors, where $\hat{x}_t = [\hat{x}_t^1, \hat{x}_t^2, \dots, \hat{x}_t^n]$ is the

Algorithm 1 Online RK-OC-ELM: Boundary Based Approach

Input: Training set $X = (x_1, c_1), (x_2, c_2), \dots, (x_{N_0}, c_{N_0}), \dots, (x_t, c_t), \dots$

Output: Whether Target or Outlier corresponding to each sample

- 1: Pass initial set of samples X_0, R_0 to the classifier as: $\{(x_1, R_1), (x_2, R_2), \dots, (x_{N_0}, R_{N_0})\}$
// For first chunk of N_0 samples, following steps are required
- 2: Employ kernel feature mapping: $\phi_0 = \phi(X_0)$.
- 3: Output Weight β_0 for (X_0, R_0) :

$$\beta_0 \leftarrow P_0 R_0$$

$$P_0 \leftarrow \phi_0^{-1}$$

$$b \leftarrow N_0$$
- // For second chunk onwards, following steps are required
- $$\phi \leftarrow \phi_0$$

$$P \leftarrow P_0$$
- 4: **for** $i = 1$ to last chunk or block of data in X **do**
- 5: Size of chunk at the current stage = s
- 6: Calculate ϕ_v for i^{th} block by Equation (6)
- 7: Calculate $\phi_{u,v}$ for i^{th} block by Equation (7)
- 8: Update the final kernel matrix ϕ by using Equation (5)
- 9: Inverse of final kernel matrix ϕ , i.e. calculate P by using Equation (8)
- 10: Solve Equation (8) by Equation (9)-(12)
- 11: $b = b + s$
- 12: **end for**
- 13: Output Weight $\beta = PR$
- 14: Compute the predicted value by using output function $f(X_k)$, which is defined in Equation (13)
- 15: Calculate distances(d) between predicted value of training sample and R as per Equation (14)
- 16: Sort the distances in decreasing order
- 17: Compute θ_1 by Equation (15)
- 18: Use Equation (16) to decide whether a new sample z belongs to target or not

predicted output vector for the t^{th} sample. $\hat{c} = [\hat{c}_1, \hat{c}_2, \dots, \hat{c}_t, \dots]$ is the predicted class vector, where, \hat{c}_t is the predicted class for the t^{th} sample. Formulation of **Online RK-OC-ELM** for reconstruction framework based approach is as discussed next:

For initial chunk of N_0 dataset $\{X_0, X_0\}$, objective is to minimize the output weight β_0 as well as error E_0 between expected (X_0) and predicted values ($\phi(X_0)\beta_0$). Minimization problem can be written as follows:

$$\begin{aligned} \text{Minimize : } L &= \frac{1}{2} \|\beta_0\|^2 + \lambda \frac{1}{2} \|E_0\|^2 \\ \text{Subject to : } &\phi(X_0)\beta_0 = X_0 - E_0, \end{aligned} \quad (17)$$

After solving the above minimization problem using Karush-Kuhn-Tucker (KKT) theorem [33], it yields following solution:

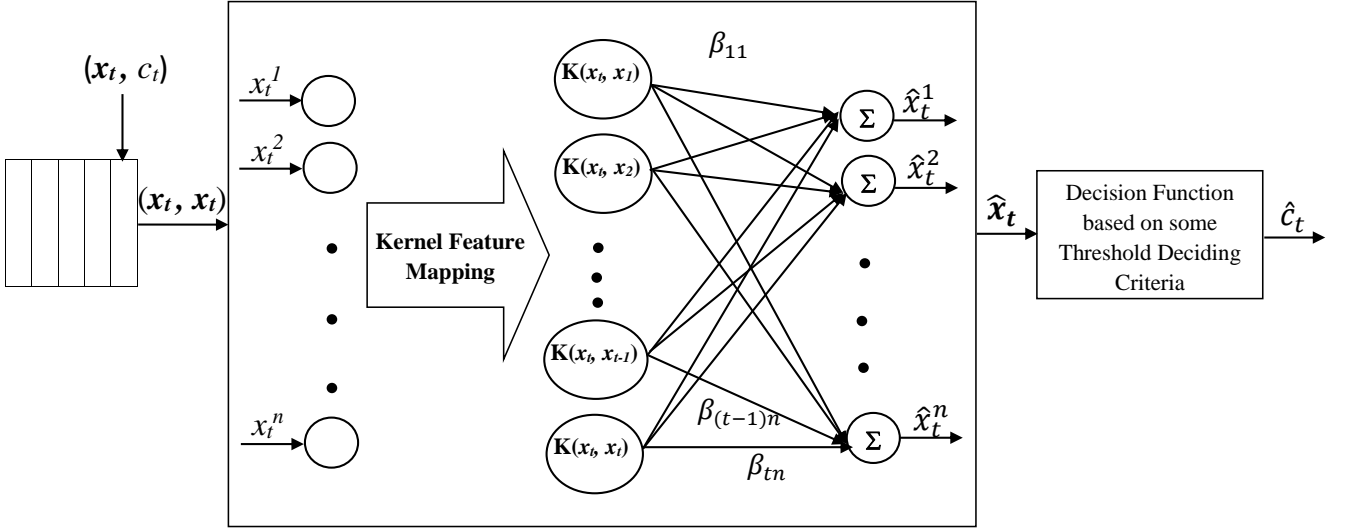


Fig. 2: Schematic Diagram of Online Sequential One Class Extreme Learning Machine: Reconstruction Based

$$\begin{aligned}\beta_0 &= P_0 X_0 \\ P_0 &= \phi_0^{-1} \\ \phi_0 &= \phi(X_0)\end{aligned}\quad (18)$$

Here, ϕ_0 is a kernel matrix for initial N_0 of size $N_0 \times N_0$.

Initially, ϕ and P will be equal to ϕ_0 and P_0 , respectively. Here, ϕ represents kernel matrix and P represents inverse of this kernel matrix for all the arrived samples till now for training. Both of these will be updated continuously for any upcoming new samples X^v as per Equation (5)-(12). After processing the training dataset X , output function for any set of k samples $X_k = \{x_1, x_2, \dots, x_k\}$ can be written as follows:

$$f(X_k) = \begin{bmatrix} K(X, x_k) \\ \vdots \\ K(X, x_1) \end{bmatrix}^T P X \quad (19)$$

where, $K(X, x_i)$ denotes kernel vector for i^{th} sample x_i .

Afterwards, calculate error and decide whether any sample belongs to target or outlier as per pseudo code discussed in **Algorithm 2**. **Algorithm 2** introduces one extra threshold decision criterion (θ_2) compared to boundary framework based approach. Since, θ_2 is based on only reconstruction error of input data, therefore, it is not applicable for boundary framework based approach.

C. Parameter Selection for ELM based One-Class Classifier

Parameter selection is always a crucial task in the case of one-class classification. Various model selection criteria have been proposed in literature for handling parameters of the kernel, however, most of them are developed especially for Gaussian kernel only [35] [36]. Among various methods, consistency based model selection [37] is only suitable for any

type of Kernels. Hence, consistency based model selection has been applied for both the frameworks based OCC. There are two parameters for kernel feature mapping with a Gaussian kernel i.e. regularization parameter (λ) and kernel parameter (σ). **Consistency based model selection** [37] also employs K-fold cross validation within it (see Equation (22) below). 5-fold cross validation is used with all one-class classifiers. This model selection works mainly based on 2-sigma bound of classification model. It determines a threshold error (E_{thr}) by using Equation (22), and executes the proposed classifier with different parameters of the classifier till it yields less error than E_{thr} .

$$\begin{aligned}E_{thr} &= (M * \eta + \sigma_{thr} * \sqrt{(\eta * (1 - \eta) * M)}) / M \\ &= \eta + \sigma_{thr} * \sqrt{(\eta * (1 - \eta) / M)}\end{aligned}\quad (22)$$

where, M is equal to (N/f) , f is number of folds, σ_{thr} is threshold require for determining decision boundary during model selection, M denotes number of samples in the validation set, η denotes the fraction of the samples rejected from the dataset which lies between 0 and 1, $(M * \eta)$ is the expected number of rejected samples, $\eta * (1 - \eta) * M$ is the variance and after square root of this is standard deviation and $(M * \eta + \sigma_{thr} * \sqrt{(\eta * (1 - \eta) * M)})$ is the maximum allowed number of rejected target objects.

The above discussed model selection selects the optimal value of the parameter. If the classifier has more than one parameters then this will search for the optimal value of every parameter on all possible combinations of range of the value of the parameters. Since, proposed classifiers have two parameters σ and λ and the value of these parameters are decided by this model selection criteria, which performs a linear search on the range of all possible combinations of σ and λ . When the optimal consistent boundary is obtained then further search will be stopped. Here, among regularization parameter (λ) and kernel parameter (σ), σ is given a higher

Algorithm 2 Online RK-OC-ELM:Reconstruction Based Approach**Input:** Training set $\mathbf{X} = \{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \dots, (\mathbf{x}_{N_0}, c_{N_0}), \dots, (\mathbf{x}_t, c_t), \dots\}$ **Output:** Whether Target or Outlier corresponding to each sample

- 1: Pass initial set of samples \mathbf{X}_0 to the classifier as: $\{(\mathbf{x}_1, \mathbf{x}_1), (\mathbf{x}_2, \mathbf{x}_2), \dots, (\mathbf{x}_{N_0}, \mathbf{x}_{N_0})\}$
 // For first chunk of N_0 samples, following steps are required
- 2: Employ kernel feature mapping: $\phi_0 = \phi(\mathbf{X}_0)$.
- 3: Output Weight β_0 for $(\mathbf{X}_0, \mathbf{X}_0)$:

$$\begin{aligned}\beta_0 &\leftarrow P_0 \mathbf{X}_0 \\ P_0 &\leftarrow \phi_0^{-1} \\ b &\leftarrow N_0\end{aligned}$$

// For second chunk onwards, following steps are required

$$\begin{aligned}\phi &\leftarrow \phi_0 \\ P &\leftarrow P_0\end{aligned}$$

- 4: **for** $i = 1$ to last chunk or block of data in \mathbf{X} **do**
- 5: Size of chunk at the current stage = s
- 6: Calculate ϕ_v for i^{th} block by Equation (6)
- 7: Calculate $\phi_{u,v}$ for i^{th} block by Equation (7)
- 8: Update the final kernel matrix ϕ by using Equation (5)
- 9: Inverse of final kernel matrix ϕ , i.e. calculate P by using Equation (8)
- 10: Solve Equation (8) by Equation (9)-(12)
- 11: $b = b + s$
- 12: **end for**
- 13: Output Weight $\beta = P\mathbf{X}$
- 14: Compute the predicted value by using output function $f(\mathbf{X}_k)$, which is defined in Equation (19)
- 15: **if** Threshold deciding Criteria is θ_1 **then**
- 16: Calculate sum of square error (d) between predicted value ($\hat{\mathbf{x}}_t$) of training sample and actual value (\mathbf{x}_t) of t^{th} sample as follows:

$$d(\mathbf{x}_t) = \sum_{j=1}^n (\mathbf{x}_t^j - \hat{\mathbf{x}}_t^j)^2 \quad (20)$$

- 17: Sort the distances in decreasing order
- 18: Compute θ_1 by Equation (15)
- 19: Use Equation (16) to decide whether a new sample z belongs to target or not
- 20: **else if** Threshold deciding Criteria is θ_2 **then**
- 21: Calculate relative error (d) between predicted ($\hat{\mathbf{x}}_t^j$) and actual value (\mathbf{x}_t^j) for j^{th} attribute of t^{th} sample as follows:

$$d(\mathbf{x}_t^j) = abs \left(\frac{\mathbf{x}_t^j - \hat{\mathbf{x}}_t^j}{\mathbf{x}_t^j + \hat{\mathbf{x}}_t^j} \right) * 100 \quad (21)$$

- 22: For any t^{th} sample with n attributes, Number of properly reconstructed attributes ($prop_reconst$)=0
- 23: **for** $j=1$ to n **do**
- 24: **if** $d(\mathbf{x}_t^j) \leq 50\%$ **then**
- 25: $prop_reconst \leftarrow prop_reconst + 1$
- 26: **end if**
- 27: **if** $prop_reconst \geq \lfloor 0.9 * n \rfloor$ **then**
- 28: t^{th} sample is target
- 29: **else**
- 30: t^{th} sample is Outlier
- 31: **end if**
- 32: **end for**
- 33: **end if**

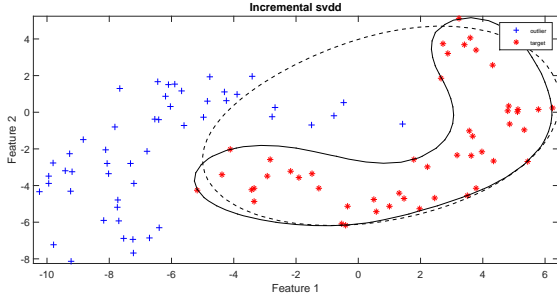


Fig. 3: Class boundaries are estimated at different values of σ in Incremental SVDD. Dashed line boundary at $\sigma = 9$ and complete line boundary at $\sigma = 4$ for Banana dataset.

TABLE I: Dataset Description

Dataset Name	# Attributes	# Targets	# Outliers	Name of Target Class
Breast Cancer	9	241	458	Benign
Diabetes	8	500	268	Present
Ecoli	7	52	284	Periplasm
Liver	6	145	200	Healthy
Sonar	60	111	97	Mines
Spectf	44	95	254	0

priority compared to λ . Suppose the consistent boundary is obtained at two different parameter combinations, (σ_1, λ_1) and (σ_2, λ_2) then smaller σ is preferred over larger λ as changing in small amount of sigma impacts more compared to small amount of change in regularization parameter. Every possible combination of σ and λ have been employed to obtain most complex classifier as long as classifier is consistent. Range of λ taken is $[10^{-8}, 10^{-7}, \dots, 10^7, 10^8]$ and range of σ is taken as twenty values between minimum and maximum pairwise distance among training samples of the dataset. Further, just for showing the impact of appropriate parameter selection, two boundaries are exhibited in Fig. 3 on banana-shaped dataset for incremental SVDD. Dashed line shows the boundary without optimal parameter selection and complete line boundary shows the boundary using the optimal parameter obtained by consistency based model selection.

III. PERFORMANCE EVALUATION

In this section, first discuss the performance of the proposed classifiers on two artificial datasets and then discuss performance on six benchmark datasets. All experiments for this paper have been executed on MATLAB 2011a in Windows 7 (64 bit) environment with 4 GB RAM, 3.10 GHz processor and Intel i5 processor. For implementing the existing classifiers, two toolboxes is used, which is proposed by Tax [10] and Gautam [30].

A. Performance Comparison on Artificial Datasets

In this section, the impact of online regularized kernel on boundary creation for two artificial datasets is discussed. Artificial datasets are created with the help of PRTToolBox [38] and each dataset is of size hundred. Boundary is created around

artificial dataset with and without kernel feature mapping. Without kernel feature mapping indicates that random feature mapping is employed instead of kernel feature mapping. Just for sake of notation we have given name to the one-class classifier without kernel feature mapping as online regularized one-class ELM (**Online R-OC-ELM**). **Online R-OC-ELM** is discussed for the both frameworks in the supplementary material provided with this paper. Further, impact of threshold on boundary creation for artificial dataset is discussed. Classifier name with employed threshold criteria is denoted as *classifier_threshold* in this paper. For e.g., **online RK-OC-ELM** $_{\theta_1}$ and **online RK-OC-ELM** $_{\theta_2}$ denote that **online RK-OC-ELM** employed θ_1 and θ_2 respectively as threshold deciding criteria. Same naming convention for classifiers' name have been used during whole discussion now onwards in **Section III** and **Section IV**. Hence, notation of existing ELM based classifier is also changed from their paper based on which threshold criteria is employed as follows:

- ELM based offline one-class classifiers with kernel feature mapping: OCKELM $_{\theta_1}$ is boundary framework based approach. AAKELM $_{\theta_1}$ and AAKELM $_{\theta_2}$ are reconstruction framework based approach
- ELM based online one-class classifiers with random feature mapping: OS-OC-ELM $_{\theta_1}$ is boundary framework based approach. OS-AAELM $_{\theta_1}$ and OS-AAELM $_{\theta_2}$ are reconstruction framework based approach.

1) Impact of Threshold on Boundary Creation for Artificial Datasets: Threshold deciding criterion is a very crucial factor in one-class classification. We employ one threshold deciding criterion θ_1 with boundary framework based one-class classifier and two threshold deciding criteria viz., θ_1 and θ_2 , with reconstruction framework based one-class classifier. Any one threshold deciding criterion is employed after the output obtained at the last layer of the one-class classifiers. The ability of the proposed methods on boundary creation has been tested on two artificial datasets viz., Banana-shaped and Ring-shaped datasets. These are used to test the ability of the proposed one-class classifiers for convex and internal boundary creation. Convexity of the proposed methods is evaluated based on the boundary creation ability around Banana shaped dataset. Fig. 4 exhibits the behavior of the proposed one-class classifiers on creation of convex decision boundary around banana-shaped dataset. One more thing can be noticed in Fig. 4e and 4f that they don't reject any sample during boundary creation due to threshold deciding criteria (θ_2). However, classifiers with threshold θ_1 reject 10% samples (can be seen from Fig. 4a-4d).

Second artificial dataset is Ring-shaped dataset. Proposed classifiers are tested on this dataset to test the ability of classifiers to create an internal boundary of the ring because internal boundary differentiates it from the only circular boundary. If parameters are not chosen appropriately, then classifier may create only outer circular boundary of the ring as it covers all samples within it but it also covers extra void space within it. It is experienced that the performance of all methods over Ring-shaped dataset is similar to Banana dataset except **online R-OC-ELM** $_{\theta_2}$ (same can be verified

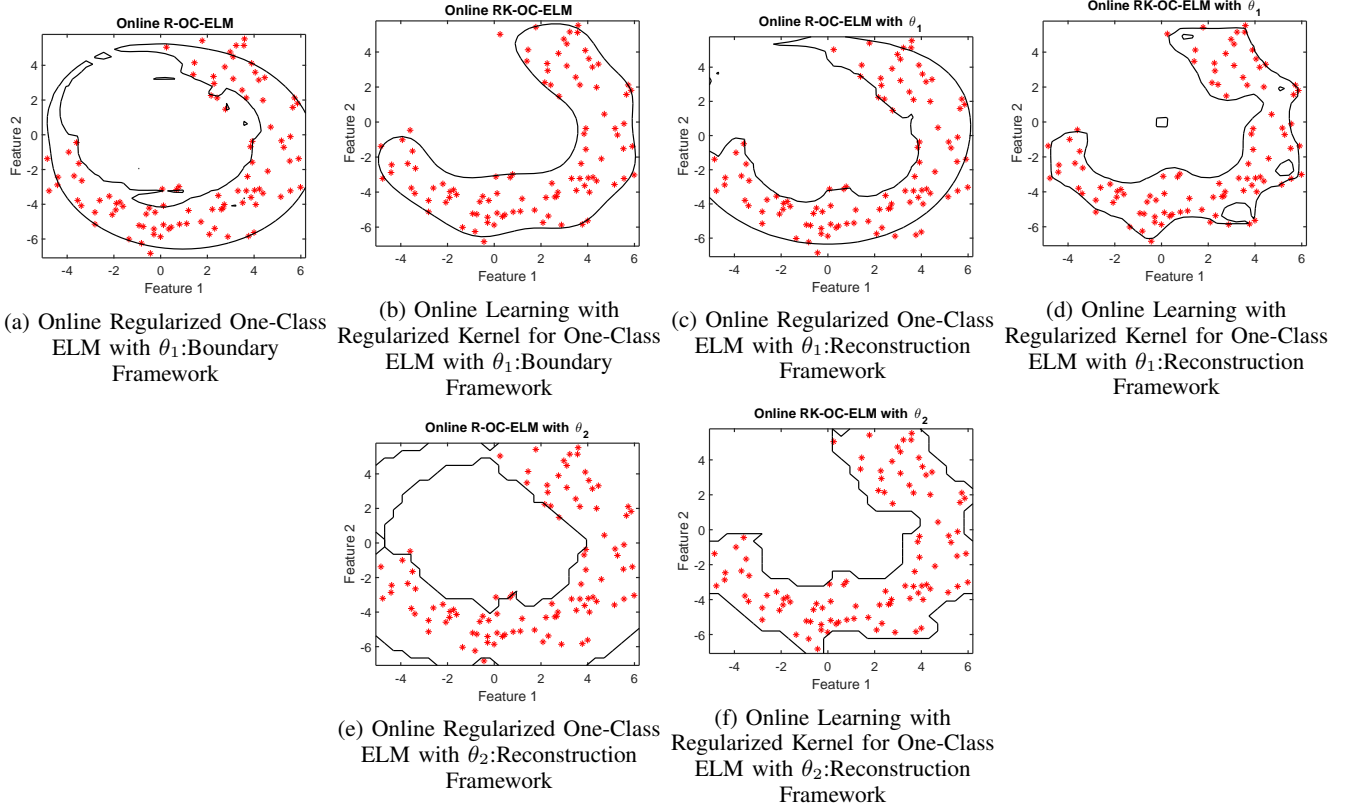


Fig. 4: Performance of Proposed Online Classifiers on Synthetic Banana dataset:Boundary and Reconstruction Framework based approaches

in Fig. 5). **Online R-OC-ELM** $_{\theta_2}$ classifier is not able to create boundary around ring dataset as you can see in Fig. (5e). However, in contrast, when **RK-OC-ELM** is employed with θ_2 (**online RK-OC-ELM** $_{\theta_2}$) then performance was just opposite, it creates boundary perfectly without rejecting any amount of sample outside of its boundary. This classifier is tested **online RK-OC-ELM** with various other artificial datasets to check its boundary creation ability and found the same performance. By above discussion, it is clear that θ_2 is more suitable for reconstruction framework based approach.

2) **Impact of Online Regularized Kernel on Boundary Creation for Artificial Datasets:** In Fig. 4 and 5, with and without online kernel feature mapping based methods (i.e. **online RK-OC-ELM** and **online R-OC-ELM**, respectively) are kept side by side for better visualization. As you can see in Fig. 4, just adding regularization factor without kernel feature mapping (i.e. **online R-OC-ELM**) does not help much in boundary creation as they are not able to create smooth boundary for Banana dataset as well as they cover some extra void space. It means that data is not described properly by one-class classifiers. However, when kernelized feature mapping is embedded instead of random feature mapping and tested the proposed classifier **online RK-OC-ELM** on Banana dataset then performance is significantly improved. **online RK-OC-ELM** based one-class classifiers create proper and smooth boundaries around Banana dataset. Overall two points are note here: (i) The proposed classifiers achieve similar boundary cre-

ation ability as offline kernel feature mapping based classifiers presented by Gautam et al. [30] (ii) Performance of kernel feature mapping is superior over random feature mapping, therefore, it was necessary to implement it with online ELM based one-class classifiers.

B. Performance Comparison on Benchmark Datasets

The performance of the proposed methods have been tested on six benchmark datasets as mentioned in Table I. Benchmark datasets are available at UCI Machine Learning Repository [39] in original format and available in one-class format on the website of the TU Delft [40]. The datasets are prepared for OCC in the same way as prepared by Leng et al. [28] and Gautam et al. [30]. Dataset is divided according to their classes and assume normal samples as target class and remaining of the classes as outlier class. And same procedure is followed for each dataset. Further, randomly select 50% of target data for training and the remaining 50% is combined with the outlier class for testing. It is to be noted that optimal parameter is calculated in the first run only. Hence, it speeds up the execution and saves a lot of time. Data is normalized between 0 and 1 using max-min normalization. Classifiers are experimented over twenty runs and then calculate average of accuracy (ACC), area under curve ($AUC = 1/2(\text{Sensitivity} + \text{Specificity})$) and F-measure ($F1$) over all runs as final output. All three performance measures are presented in the result tables (Table II-VII) but, AUC is considered as a performance

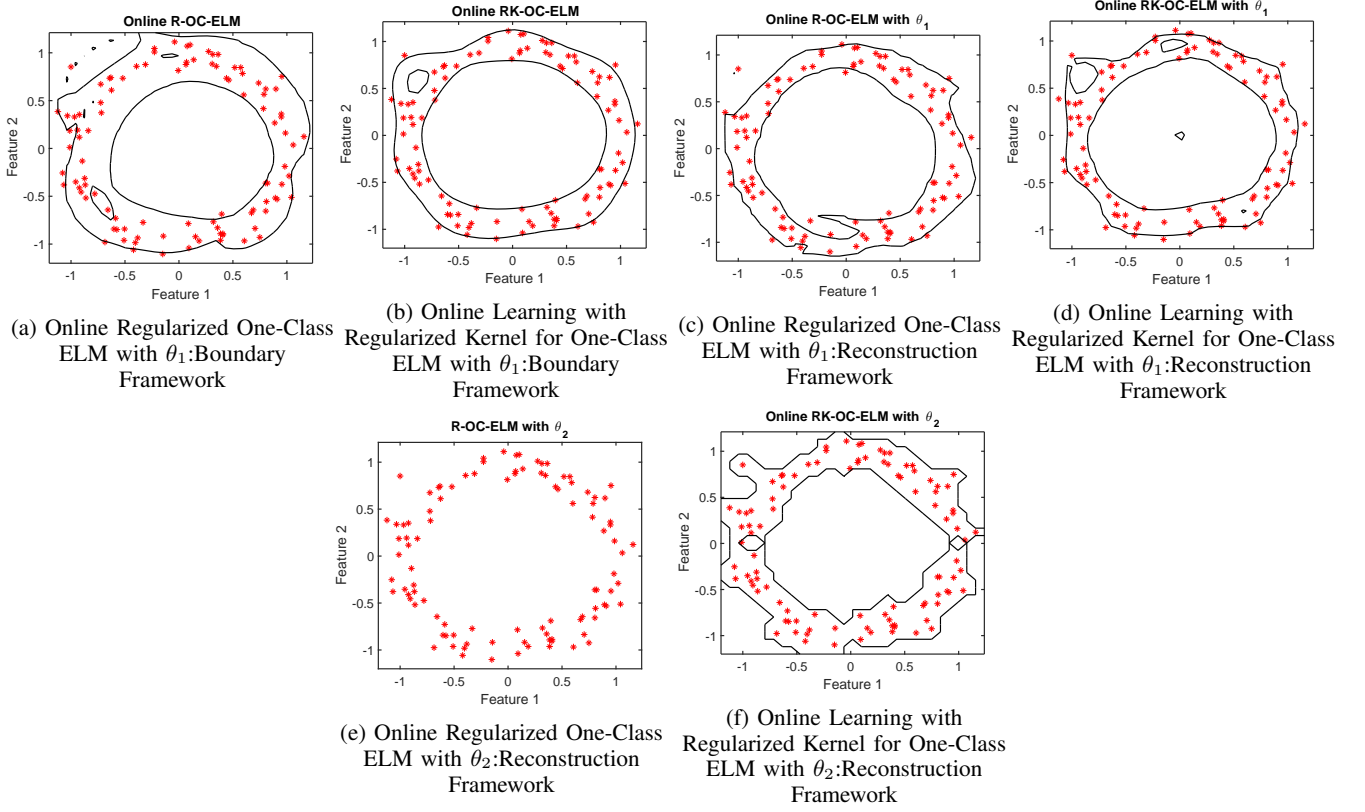


Fig. 5: Performance of Proposed Online Classifiers on Synthetic Ring dataset:Boundary and Reconstruction Framework based approaches

TABLE II: Boundary Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Breast Cancer and Ecoli Dataset (Well separated classes)

Methods	Breast Cancer			Ecoli		
	F1	ACC	AUC	F1	ACC	AUC
incsvdd [6]	94.18 (1.2)	94.39 (1.13)	94.36 (1.14)	39.18 (19.27)	72.9 (20.17)	76.39 (13.43)
OCKELM_ θ_1 [24]	93.22 (1.89)	93.84 (1.6)	93.68 (1.64)	77.04 (5.96)	96.35 (0.82)	86.04 (4.13)
OS-OCELM_ θ_1 [26]	86.96 (2.19)	87.73 (2.25)	87.64 (2.21)	34.68 (8.43)	74.35 (11.34)	73.77 (5.32)
Online RK-OC-ELM_θ_1	94.97 (1.55)	95.34 (1.35)	95.22 (1.39)	71.36 (5.05)	94.56 (1.24)	88.12 (4.57)

TABLE III: Reconstruction Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Breast Cancer and Ecoli Dataset (Well separated classes)

Methods	Breast Cancer			Ecoli		
	F1	ACC	AUC	F1	ACC	AUC
autoenc_dd [6]	92.54 (1.75)	93.2 (1.49)	93.05 (1.52)	46.61 (13.28)	82.79 (9.64)	80.39 (6.81)
AAKELM_ θ_1 [26]	93.53 (1.6)	94.1 (1.38)	93.94 (1.41)	20.31 (1.38)	40.82 (2.29)	63.25 (3.5)
AAKELM_ θ_2 [26]	94.7 (0.93)	95.11 (0.81)	94.98 (0.83)	56.98 (7.78)	90.15 (3.35)	83 (3.5)
OS-AAELM_ θ_1 [26]	90.41 (1.78)	91.49 (1.43)	91.27 (1.47)	50.83 (15.56)	85.47 (9.09)	80.54 (5.37)
OS-AAELM_ θ_2 [26]	89.96 (2.79)	89.55 (3.23)	89.69 (3.14)	54.01 (9.92)	90.58 (4.1)	78.09 (5.67)
Online RK-OC-ELM_θ_1	93.2 (1.2)	93.79 (1.03)	93.63 (1.05)	20.03 (2.25)	41.11 (2.22)	62.62 (5.27)
Online RK-OC-ELM_θ_2	94.6 (0.86)	95.01 (0.75)	94.88 (0.77)	55.54 (8.89)	88.52 (3.72)	85.26 (2.9)

TABLE IV: Boundary Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Diabetes and Liver Dataset (Overlapping classes)

Methods	Diabetes			Liver		
	F1	ACC	AUC	F1	ACC	AUC
incsvdd [6]	66.51 (0.73)	55.98 (1.5)	57.15 (1.43)	41.33 (1.34)	32.17 (0.91)	50.79 (1.63)
OCELM_ θ_1 [24]	68.86 (1.31)	63.38 (1.28)	64.07 (1.28)	42.18 (1.59)	42.06 (1.77)	54.16 (2.03)
OS-OCELM_ θ_1 [26]	61.66 (2.15)	59.16 (2.52)	59.46 (2.45)	41.68 (2.38)	48.66 (4.35)	55.31 (2.57)
Online RK-OC-ELM_θ_1	69.32 (0.89)	63.36 (0.92)	64.11 (0.91)	43.81 (1.81)	45.83 (1.45)	56.72 (2.05)

TABLE V: Reconstruction Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Diabetes and Liver Dataset (Overlapping classes)

Methods	Diabetes			Liver		
	F1	ACC	AUC	F1	ACC	AUC
autoenc_dd [6]	66.27 (1.49)	57.48 (2.65)	58.46 (2.54)	42.26 (2.03)	38.9 (2.96)	53.52 (2.51)
AAKELM_ θ_1 [26]	67.39 (0.7)	60.52 (0.98)	61.33 (0.89)	41.21 (2.03)	38.07 (2.41)	52.24 (1.59)
AAKELM_ θ_2 [26]	59.01 (1.67)	59.48 (1.58)	59.51 (1.58)	42.36 (1.38)	44.21 (1.65)	54.86 (1.62)
OS-AAELM_ θ_1 [26]	63.46 (2.19)	63.23 (1.71)	63.34 (1.62)	38.92 (4.68)	55.64 (5.85)	55.3 (3.32)
OS-AAELM_ θ_2 [26]	60.19 (3.64)	55.72 (3.15)	56.22 (2.76)	38.16 (5.54)	57.19 (6.33)	55.47 (2.5)
Online RK-OC-ELM_θ_1	67.16 (0.45)	60.45 (0.99)	61.24 (0.87)	41.03 (1.95)	38.31 (2.92)	52.12 (1.41)
Online RK-OC-ELM_θ_2	58.44 (1.89)	59.37 (1.39)	59.37 (1.42)	42.63 (1.37)	43.22 (1.79)	54.9 (1.72)

TABLE VI: Boundary Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Sonar and Spectf Dataset (High dimensional datasets)

Methods	Sonar			Spectf		
	F1	ACC	AUC	F1	ACC	AUC
incsvdd [6]	57.89 (2.58)	55.59 (2.48)	61.84 (2.79)	46.2 (2.26)	68.12 (4.43)	75.87 (1.88)
OCELM_ θ_1 [24]	56.44 (3.39)	64.18 (3.02)	64.16 (3.06)	53.75 (3.44)	80.35 (1.43)	77.48 (3.17)
OS-OCELM_ θ_1 [26]	54.5 (5.46)	61.25 (5.22)	62.02 (4.33)	30.91 (7.54)	52.87 (14.19)	57.25 (7.54)
Online RK-OC-ELM_θ_1	58.75 (4.3)	70.23 (2.99)	67.74 (3.28)	57 (3.81)	82.76 (2.13)	78.95 (3.8)

measure criterion for further discussion as AUC provides better insight of classifiers' performance compared to ACC and $F1$. Standard deviation of $F1$, ACC and AUC over twenty runs is also presented to get a proper understanding of possible deviation in the performance over twenty runs.

The proposed and existing ELM based online classifiers have four parameters if use RBF kernel viz., the number of initial training data used in the initial phase of the online sequential one-class classifier, size of blocks of data learned by online sequential one-class classifier in each iteration, regularization parameter (λ) and kernel parameter (σ). Out of four parameters, values of two parameters are fixed and values for the rest of the two parameters (λ and σ) have been decided by consistency based model selection. The value of these two fixed parameters are taken as 10 and 5 for number of initial training data used in the initial phase and size of blocks of data learned by the classifier in each iteration, respectively.

In this section, first discuss performance comparison of boundary framework based one-class classifiers and then provide discussion on reconstruction framework based one-class classifiers.

1) Performance Comparison of Boundary Framework Based Methods on Benchmark Datasets: Threshold cri-

terion θ_1 has been employed in all existing and proposed approaches presented in the the Table II, IV and VI. We compare the proposed classifiers with one of the most popular traditional methods, incremental SVDD and recently proposed ELM based online and offline one-class classifiers. Proposed classifier **online RK-OC-ELM_ θ_1** is the online version of the existing OCKELM_ θ_1 [28] classifier. It is expected that **online RK-OC-ELM_ θ_1** will yield similar results as OCKELM_ θ_1 . As you can see from the table, the proposed classifier **online RK-OC-ELM_ θ_1** outperforms all existing online as well as offline classifier for all six datasets in term of AUC . It is observed that proposed kernel feature mapping based method **online RK-OC-ELM_ θ_1** performed better than existing random feature mapping based online method viz., OS-OCELM_ θ_1 [30]. Superiority of kernel over random feature mapping was expected because Leng et al. [28] and [30] had also exhibited the same trend in their work for offline classifiers.

2) Performance Comparison of Reconstruction Framework Based Methods on Benchmark Datasets: Proposed classifier **online RK-OC-ELM** has been employed with two threshold criteria viz., θ_1 and θ_2 (i.e. **online RK-OC-ELM_ θ_1** and **online RK-OC-ELM_ θ_2**). These are the online version of the existing offline classifiers AAKELM_ θ_1 and

TABLE VII: Reconstruction Framework: Performance of Various One-class Classifiers over Average of 20 Runs with their Standard Deviation in Bracket on Sonar and Spectf Dataset (High dimensional datasets)

Methods	Sonar			Spectf		
	F1	ACC	AUC	F1	ACC	AUC
autoenc_dd [6]	48.23 (2.91)	38.65 (4.02)	47.41 (3.63)	50.4 (2.92)	76.58 (3.25)	76.37 (3.15)
AAKELM_ θ_1 [26]	49.16 (1.63)	36.22 (1.27)	46.84 (1.83)	59.24 (4.23)	87.38 (1.02)	75.87 (3.21)
AAKELM_ θ_2 [26]	62.02 (5.55)	75.89 (2.8)	71.33 (3.62)	57.88 (3.17)	80.8 (1.01)	82.42 (3.39)
OS-AAELM_ θ_1 [26]	56.11 (4.35)	66.28 (5.6)	64.8 (4.29)	32.3 (9.69)	48.55 (17.95)	57.55 (8.42)
OS-AAELM_ θ_2 [26]	39.91 (18.46)	68.42 (3.64)	60.89 (6.98)	NaN	84.57 (0.42)	50.59 (1.36)
Online RK-OC-ELM_θ_1	49.77 (2)	36.81 (1.29)	47.6 (2.09)	59.27 (4.67)	87.87 (0.99)	75.26 (3.28)
Online RK-OC-ELM_θ_2	63.25 (3.17)	72.93 (2.28)	71.11 (2.49)	58.45 (2.34)	80.91 (0.95)	83.01 (2.36)

AAKELM_ θ_2 , respectively. Therefore, it is expected that both online and their corresponding offline versions will exhibit similar performance and these facts can be verified by the results of the tables (Table III, V and VII). It can also be easily inferred from these tables that θ_2 performs better than θ_1 since θ_2 does not decide threshold by rejecting any amount of samples. Except for Diabetes and Liver dataset, the proposed classifiers outperform all existing online one-class classifiers presented in the table significantly in term of *AUC*. Even, there is no significant difference in the value of *AUC* between OS-AAELM_ θ_3 and **online RK-OC-ELM_ θ_2** . Therefore, it can be stated that kernel feature mapping based classifiers have again exhibited the superiority over random feature mapping based classifiers. It is observed that θ_2 based method **online RK-OC-ELM_ θ_2** has been performed better than θ_1 based method **online RK-OC-ELM_ θ_1** for most of the datasets, hence, **online RK-OC-ELM_ θ_2** should be preferred over **online RK-OC-ELM_ θ_1** for reconstruction framework based approaches. But, threshold criteria θ_2 exhibits strange behaviour for random feature mapping based classifier as it can be seen in Table VII. Since, the proposed works are based on kernel feature mapping, hence we didn't encounter that issue but, it is not suggested to use θ_2 with random feature mapping based classifier.

C. Discussion

In the previous section, the performance of the proposed classifiers has been discussed for boundary and reconstruction frameworks based approaches separately. This section will provide the general discussion on both frameworks together. Table II-VII present all results obtained on six benchmark datasets. It can be seen in the tables that proposed online one-class classifiers achieve similar performance as ELM based offline kernelized one-class classifiers (i.e. OCKELM_ θ_1 , AAKELM_ θ_1 , AAKELM_ θ_2). However, it is not necessary that proposed online classifiers will achieve exactly the same output as offline classifier. It is due to two reasons: First, due to selection of training and testing data randomly in each run for any classifier. Second, method of inverse calculation is different in online and offline classifiers.

The proposed classifiers have been tested with various benchmark datasets and it is found that reconstruction based one-class classifiers perform better than boundary framework

based methods for 4 out of 6 datasets, only outperformed for Breast Cancer and Ecoli datasets. Here also, there is no significant difference between the performance of reconstruction framework based and boundary framework based classifiers for the Breast Cancer dataset. Two types of threshold viz., θ_1 and θ_2 have been employed with reconstruction framework based approach but only one type of threshold θ_1 for boundary framework based approach. θ_2 is not employed for boundary framework based classifier as it works based on reconstruction of data at output layer.

IV. CONCLUSION

This paper has presented online learning with regularized kernel for one-class classification. Here, regularization parameter with kernel helps the classifier to achieve good generalization capability. Three methods have been discussed hence far in the paper, which are based on boundary and reconstruction frameworks. **online RK-OC-ELM_ θ_1** has been developed for boundary and reconstruction frameworks both, however, **online RK-OC-ELM_ θ_2** has been developed only for reconstruction framework based as θ_2 cannot be employed for boundary framework based classifier.

Our proposed classifier can handle data in an online manner and performance evaluation of these classifiers have exhibited that these online classifiers are equally capable as offline classifiers. As the proposed classifiers are online, hence, there is no need to train the model from scratch for any new arrival of training data. Simply, pass newly arrived data with the proposed classifiers and this will take care of the rest. However, proposed classifiers have been tested on only small size of benchmark datasets but, since these are an online classifier, hence, they are capable of handling large size dataset also. These online classifiers either outperformed existing online one-class classifier for most of the dataset or yielded similar results for some of the datasets. Overall, reconstruction framework based one-class classifiers have performed better compared to boundary framework based one-class classifiers.

Consistency based model selection has been employed as a model selection criteria for all the existing and proposed one-class classifiers. This model selection is efficient in the case where only two free parameters are there in the classifiers. However, it would be very time consuming if there would be more than two parameters as it searches the appropriate

parameter by linear search on all possible combination of the parameters. Any evolutionary optimization technique can be employed to overcome this issue.

ACKNOWLEDGMENT

This research was supported by Department of Electronics and Information Technology (DeITY, Govt. of India) under Visvesvaraya PHD scheme for electronics & IT.

REFERENCES

- [1] M. M. Moya, M. W. Koch, and L. D. Hostetler. One-class classifier networks for target recognition applications. Technical report, Sandia National Labs., Albuquerque, NM (United States), 1993.
- [2] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.
- [3] C. Zhou, S. Huang, N. Xiong, S.-H. Yang, H. Li, Y. Qin, and X. Li. Design and analysis of multimodel-based anomaly intrusion detection systems in industrial process automation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(10):1345–1360, 2015.
- [4] C. F. Stallmann and A. P. Engelbrecht. Gramophone noise detection and reconstruction using time delay artificial neural networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, PP(99):1–13, 2016.
- [5] L. Zhang, J. Lin, and R. Karim. Sliding window-based fault detection from high-dimensional data streams. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, PP(99):1–15, 2016.
- [6] A. Mammeri, D. Zhou, and A. Boukerche. Animal-vehicle collision mitigation system for automated vehicles. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(9):1287–1299, 2016.
- [7] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *NIPS*, volume 12, pages 582–588, 1999.
- [8] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern recognition letters*, 20(11):1191–1199, 1999.
- [9] D. M. J. Tax and P. Laskov. Online SVM learning: from classification to data description and back. In *IEEE 13th Workshop on Neural Networks for Signal Processing, 2003 (NNSP'03)*, pages 499–508. IEEE, 2003.
- [10] D. M. J. Tax. DDtools, the data description toolbox for MATLAB, version 2.1.2, June 2015.
- [11] S. S. Khan and M. G. Madden. A survey of recent trends in one class classification. In *Irish conference on Artificial Intelligence and Cognitive Science*, pages 188–197. Springer, 2009.
- [12] N. Japkowicz. *Concept-learning in the absence of counter-examples: An autoassociation-based approach to classification*. PhD thesis, Rutgers, The State University of New Jersey, 1999.
- [13] L. Manevitz and M. Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7):1466–1481, 2007.
- [14] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.
- [15] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(2):513–529, 2012.
- [16] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 17(6):1411–1423, 2006.
- [17] S. Scardapane, D. Comminiello, M. Scarpiniti, and A. Uncini. Online sequential extreme learning machine with kernels. *IEEE Transactions on Neural Networks and Learning Systems*, 26(9):2214–2220, 2015.
- [18] X. Wang and M. Han. Online sequential extreme learning machine with kernels for nonstationary time series prediction. *Neurocomputing*, 145:90–97, 2014.
- [19] X. Zhou, Z. Liu, and C. Zhu. Online regularized and kernelized extreme learning machines with forgetting mechanism. *Mathematical Problems in Engineering*, 2014:1–11, 2014.
- [20] Z. Bai, G.-B. Huang, D. Wang, H. Wang, and M. B. Westover. Sparse extreme learning machine for classification. *IEEE Transactions on Cybernetics*, 44(10):1858–1870, 2014.
- [21] H. Zhou, G.-B. Huang, Z. Lin, H. Wang, and Y. C. Soh. Stacked extreme learning machines. *IEEE Transactions on Cybernetics*, 45(9):2013–2025, 2015.
- [22] A. Iosifidis, A. Tefas, and I. Pitas. Graph embedded extreme learning machine. *IEEE Transactions on Cybernetics*, 46(1):311–324, 2016.
- [23] R. Savitha, S. Suresh, and H. J. Kim. A meta-cognitive learning algorithm for an extreme learning machine classifier. *Cognitive Computation*, 6(2):253–263, 2014.
- [24] L. Zhang and D. Zhang. Domain adaptation extreme learning machines for drift compensation in e-nose systems. *IEEE Transactions on Instrumentation and Measurement*, 64(7):1790–1801, 2015.
- [25] S. Suresh, S. Saraswathi, and N. Sundararajan. Performance enhancement of extreme learning machine for multi-category sparse data classification problems. *Engineering Applications of Artificial Intelligence*, 23(7):1149–1157, 2010.
- [26] S. Suresh, R. Venkatesh Babu, and H. J. Kim. No-reference image quality assessment using modified extreme learning machine classifier. *Applied Soft Computing*, 9(2):541–552, 2009.
- [27] G. Huang, G.-B. Huang, S. Song, and K. You. Trends in extreme learning machines: a review. *Neural Networks*, 61:32–48, 2015.
- [28] Q. Leng, H. Qi, J. Miao, W. Zhu, and G. Su. One-class classification with extreme learning machine. *Mathematical Problems in Engineering*, pages 1–11, 2014.
- [29] A. Iosifidis, V. Mygdalis, A. Tefas, and I. Pitas. One-class classification based on extreme learning and geometric class information. *Neural Processing Letters*, pages 1–16, 2016.
- [30] C. Gautam, A. Tiwari, and Q. Leng. On the construction of extreme learning machine for online and offline one class classifier - An expanded toolbox. *Neurocomputing*, 2016 (Accepted, Download the submitted version of accepted paper from: <http://goo.gl/XhkVpE>).
- [31] C. Gautam and A. Tiwari. On the construction of extreme learning machine for one class classifier. In *Proceedings of ELM-2015 Volume 1*, pages 447–461. Springer, 2016.
- [32] A. N. Tikhonov and V. Y. Arsenin. Solutions of ill-posed problems. 1977.
- [33] Philip E Gill, Walter Murray, and Margaret H Wright. Practical optimization. 1981.
- [34] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.
- [35] Z. Xu, M. Dai, and D. Meng. Fast and efficient strategies for model selection of Gaussian support vector machine. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(5):1292–1307, 2009.
- [36] Y. Xiao, H. Wang, and W. Xu. Parameter selection of Gaussian kernel for one-class SVM. *IEEE Transactions on Cybernetics*, 45(5):941–953, 2015.
- [37] D. M. J. Tax and K.-R. Müller. A consistency-based model selection for one-class classification. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, volume 3, pages 363–366. IEEE, 2004.
- [38] R. P. W. Duin, P. Juszczak, D. Ridder, P. Paclik, E. Pekalska, and D. M. J. Tax. PRTtools, a MATLAB toolbox for pattern recognition, 2004.
- [39] M. Lichman. UCI machine learning repository, 2013.
- [40] TU Delft one-class dataset repository, Last Accessed by 30 Sep. 2016.



Mr. Chandan Gautam holds an M.Tech (IT) degree from University of Hyderabad and IDRBT, Hyderabad. He is currently a research scholar (PhD) at IIT Indore. His research interests include Data Mining, Machine learning, Soft Computing; especially, Extreme Learning Machine, Kernel Learning, Online Learning and One-Class classification.



Aruna Tiwari received the B.E. degree (Computer Engineering) in 1994 and M.E. degree (Computer Engineering) in 2000 from Shri Govindaram Sakseria Institute of Technology and Science, Indore and Ph.D. degree from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, India. She joined the Indian Institute of Technology, Indore, India, in 2012, where she is currently working as an Assistant Professor with the Department of Computer Science and Engineering. Her research interests include soft computing techniques, neural network learning al-

gorithms, genetic algorithms and evolutionary approaches. She has many publications in peer reviewed journals, international conferences, and book chapters. She is reviewer of many journals some of them are IEEE Transaction on KDE, Neurocomputing journal of Elsevier etc. Dr. Tiwari is life member of IEEE Computer Society of India and member of IEEE Evolutionary Computation Society.



Sundaram Suresh received the B.E degree in electrical and electronics engineering from Bharathiyar University in 1999, and the M.E. and Ph.D. degrees in aerospace engineering from the Indian Institute of Science, Bangalore, India, in 2001 and 2005, respectively. He was a Post-Doctoral Researcher in the School of Electrical Engineering, Nanyang Technological University from 2005 to 2007. From 2007 to 2008, he was with the INRIA-Sophia Antipolis, Nice, France as Research Fellow of the European Research Consortium for Informatics and Mathemat-

ics. He was with Korea University, Seoul, Korea, for a short period as a visiting faculty in Industrial Engineering. From January 2009 to December 2009, he was with the Department of Electrical Engineering, Indian Institute of Technology, Delhi, India, as an Assistant Professor. Since 2010, he has been an Associate Professor with the School of Computer Engineering, Nanyang Technological University. His research interest includes flight control, unmanned aerial vehicle design, machine learning, and optimization and computer vision.



Kapil Ahuja (MS: Virginia Tech, USA; PhD: Virginia Tech, USA; Postdoctoral Research Fellow: Max Planck Institute, Germany) has a varied background, including degrees in Computer Science, Mathematics, and Mechanical Engineering. Currently, he is working as an Assistant Professor in the CSE discipline at IIT Indore (India). Dr. Ahuja works on applying mathematics and computation to solve science and engineering problems. Specifically, his research are numerical linear algebra, numerical analysis, optimization, computational

intelligence, big data, and cloud computing.